Computer 291 Mini Project 1 Report

Abstract:
The program implemented SQLite in Python as a host language. It provides users with interactive access to a database storing dataset of an online auction site. The program using SQLite3 database to record and track information regarding to users, products, sales, bids, reviews, and previews.

User guide:
Once the program starts, it provides user with a login interface. Returning user may login using his or her password. New user can sign up by following the instructions. After successful login, user is able to perform the following actions, listing products, searching for sales, posting a sale, and searching for users. Once user select one of the operations, he or she will enter a sub menu which provides him or her more detailed operations.
The list products function will list all products with some active sales associated to them in descending order of the number of active sales. After all the sales are listed, the user can perform the following actions:Write a product review by providing a review text and a rating (a number between 1 and 5 inclusive); the other fields of a review record should be appropriately filled by the application. In particular, a unique review id is recorded, the review date is set to the current date and time, and the user should be recorded as the reviewer.
List all reviews of the product.
1. List all active sales associated to the product, ordered based on the remaining time of the sale, with sales to expire earlier appearing first.

Search for sales. The user should be able to enter one or more keywords and the system should retrieve all active sales that have at least one keyword in either sales description or product description (if the sale is associated with a product). Order the results in a descending order of the number distinct search keywords that appear in either sale description or product description (if the sale is associated with a product). The following actions are enabled after a sale is selected.

1. **Place a bid on the selected sale by entering an amount, and the application should record the bid in the database. The fields bid must be set to a unique number, bidder must be set to the current user, sid to the sid of the sale, and the bdate to the current system date and time. The bid amount must be greater than the current largest bid; otherwise, the bid should not be accepted. Note that the user can have multiple bids on the same sale.**
2. **List all active sales of the seller. The result should be ordered on the remaining time of the sale with sales to expire earlier appearing first.**
3. **List all reviews of the seller.**

Post a sale. The user should be prompted for a product id, a sale end date and time, a sale description, a condition, and a reserved price. The fields product id, and reserved price are optional and can be left unanswered. Once the data is entered, proper data should be stored

in the database with a unique sale id assigned to the sale and the lister set to the current user. The end date must be in future.

Search for users. The user should be able to enter a keyword and the system should retrieve all user profiles (including email, name, city) that have the keyword either in name or email. From the result set, one should be able to select a user and perform the following actions:

1. **write a review, by providing a review text and a rating (a number between 1 and 5 inclusive); the other fields of a review record including review date, reviewer and reviewee should be appropriately filled by the application.**
2. **list all active listings of the user.**
3. **list all reviews of the user.**

The program will check if the input from user contain any SQL meta words to prevent any SQL injection.

Design:
The program consisting of 9 Python scripts.
List products.py enables user to list all reviews of the product. Function list_products will select all active sales products and join the data with products to acquire the information regarding with the sale. It will also count the number of active sales and sort the results by that number. Then it returns the results as a list containing column description and data. The ui.py will then process the results to output the data.
write_preview enables user to select a product and write a review on it. It will generate the next product id and insert the review into the database.
list_reviews enables user to list all the reviews with a id. It will output the reviews. The returning data is processed using the same strategy as list_products.
list_sales enables user to list a sale. It implements cast to convert time to remaining time. And the results will be sorted by end date in ascending order.

Search sales.py enables user to search certain sales containing keywords. It takes a keyword list. It will first select sales containing the keyword and the combine it with the products which containing the keywords. Then the data will be insert into a temporary table. Then it will counting the number of keywords so as to sort the results in descending order by number of keywords contained in each sale. After then, cast is used to convert the time to remaining time. The returning data is processed using the same strategy as list_products.

In follow up.py, these functions enables users to perform more detailed actions. In display_infromation function, the table sales and reviews are joined to get the information like number of ratings, average rating, sales description, sales end date, and sales condition. The returning data is processed using the same strategy as list_products.
place bid enables the user to place a bid on one of the sales he or she selected. It will check if there exists a max bid on the sales otherwise it will set the minimum bid using the reserved price. It uses the strategy that join sales and bids together to find the related information.
List sales enables the user to list all sales, it lists the sales sid, description, and max bid

amount (if null, it will using the reserved price), then the results is cast to convert the time to remaining time. The results is sorted by edate in ascending order. The returning data is processed using the same strategy as list_products.
List reviews enables the user to list the reviews of selected user. It asks the user to provide a sales id to enquire the database to get the results. The returning data is processed using the same strategy as list_products.

Post sales enables the user to post a sale. The user should be prompted for a product id, a sale end date and time, a sale description, a condition, and a reserved price. The fields product id, and reserved price are optional and can be left unanswered. Once the data is entered, proper data should be stored in the database with a unique sale id assigned to the sale and the lister set to the current user. The end date must be in future. It generate the information such as pid and rprice automictically.

Search users enables the user to search any user that his or her information containing any keyword the user input. From the result set, one should be able to select a user and perform the following actions:

- **write a review, by providing a review text and a rating (a number between 1 and 5 inclusive); the other fields of a review record including review date, reviewer and reviewee should be appropriately filled by the application.**
- **list all active listings of the user.**
- **list all reviews of the user.**

COLLATE NOCASE is implemented to make the input case insensitive. The returning data is processed using the same strategy as list_products.
Then the use can write_review, providing he reviewee email and rtext and rating. It will be inserted into the database. The reviewer is pass by ui.py using current user.
The user can also list all active sales of any using providing his or her email, it will selected all the active sales and join the users with sales to get the information. The returning data is processed using the same strategy as list_products.
list reviews enables the user to list all reviews of a selected user, it needs the email, then it will enquire the data to get the information regarding to the user. The returning data is processed using the same strategy as list_products.

Injection detection.py implements regular expression to check if the password, email, and rating text contains any meta word to prevent any SQL injection.

login.py provides the user with the login interface. It will check if all constrains are met and enables new user to sign up with a unique user email. It will print the error message if any constrains are not met.

ui.py provides the user the interface to input and format the results using functions center. It also implements return after every function to return to the main menu. It provides users with the interactive access with the database underlying.

main.py is the driver function. It imports sqlite3 to connect to the certain database and get the cursor for the other functions to perform actions on the dataset.

Testing:
User input:
we tested on SQL injection by input meta word in the program.
we also input data that are out of bounds like long strings to test the program to see if it will check the constrains.
we also input the repeated user name to test if the program will crushed down.
Menu:
we tested the menu function by using it like a common user to see if the menu works.
Data:
We made our own dataset and tested in different cases to make sure the results is correct. We will calculate the result manually and then compare the results with the program output. And try different datasets, different time, to test our code.

Group work strategy:
Qianqiu contributes to question3, questions4, question5
Hao contributes to question1, question2, UI, login interface, report
Fatima contributes to SQL injection detection, data set making, calculation manually, code testing, debugging.
According to self-report, Qianqiu spent 10 hours on project. Hao spent 9 hours on the project. Fatima spent 8 hours on the project.
Qianqiu is the team leader and coordinator. She consistently monitors the progress of the project and organizing meetings to coordinate the work break down when there is a need. Github is implemented in the project as both version control system and coordinating tool.